2023蓝帽杯半决赛WriteUp

Pwn

flagserver

Write Up

程序是一个简易的文件服务器，接收客户端的指令并且把对应的文件返回给客户端。其中对路径做了处理，防止直接路径穿越读到flag

服务器循环接收客户端的数据包，数据包的格式为：

```
struct Packet{
//size.
uint32\_t body\_size;   //body
uint8\_t token;
char data[];
}
```

接收完size之后，然后动态分配一块内存去储存收到的数据接着进入on_message函数去对消息进行处理：

在每一次获取文件内容之前，都需要先发送FILE_GET命令设置文件名称。

接着才允许发送GET_FILE_CHUNK 获取文件内容：

本程序使用的是自己实现的一个简易的分配器，每一个chunk的大小固定为0x1000,一共有0x100个chunk 。

为了防止使用未初始化的数据作为ClientCtx，绕过FILE_GET 过程，直接发送GET_CHUNK命令，在alloc之前对缓冲区都做了清空。

漏洞点：

在FILE_GET命令发送之后，会调用该函数去对文件路径做处理，然后在字符串前面加上WORK_DIR，再储存到path中。

但是要注意的是，加上WORK_DIR会超过ClientCtx →filename成员的大小，由于一个ClientCtx是直接使用的0x1000的chunk，这个看起来是没什么问题的。

但是注意看在on_file_get函数中，使用的是memcpy 将buffer中的数据拷贝到了ClientCtx→filename中。如果我们先发送了一个正常的字符串，这个字符串长度满足加上WORD_DIR之后大于filename结构体。那么我们再一次发送FILE_GET命令，并且数据中(长度要大于0x100)不包含0x00，接着就会继续调用normalize_path 函数，会使filename字符串长度继续增大。反复多次，就能溢出到下一个chunk。(这个过程中需要利用memmove以及strcpy函数，需要合理布置数据才能达到目的)

在一开始建立两个连接，它们的chunk是相邻的。然后利用上面的漏洞将连接2的ClientCtx中的filename写为“/flag”并且将flag变量写为1，fd写为正常的文件描述符，然后再使用连接2直接发送get_chunk命令就能拿到flag了。

Web

MyLinuxBot

这道题考察解题者一下技能：

- 源码审计的能力
- Java经典安全问题，log4j漏洞

尝试题目的功能

题目主页是一个对话机器人，输入任意内容提示需要/作为开头执行命令,执行/ls命令发现不被允许，于是尝试help发现可以执行下述三个免费命令

```
/rp nihao
/time
/w 1 2 3
```

源码审计

在查看源代码时，我们会看到以下内容：

- Python 3.8.10在前端运行Flask web服务器
- Java 11.0.15与Log4j 2.17.2在后端

在此版本中，默认情况下禁用JNDI查找，并且Log4Shell不可利用。

查看python app.py源码，就是可以接受POST请求，提交我们text指定的命令让，然后通过调用后端的java进程处理。

分析Java部分源码发现：

- flag隐藏在环境变量flag中
- LOGGER.info是唯一一个Log4j的调用！
- 它将打印完整的参数，但不打印命令
- 从Python传递参数的方式是，只有零个或一个参数。但是，这个参数可能包含空格。
- 该命令是从系统属性cmd中获取的
- 命令必须以/开头
- 接下来，将命令和参数传递给doCommand函数

仔细研究发现题目给的三个免费命令无法利用，因此我们尝试找Log4j调用的漏洞。

分析XML文件：

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<Configuration status="INFO">
    <Appenders>
        <Console name="Console" target="SYSTEM_ERR">
            <PatternLayout pattern="%d{HH:mm:ss.SSS} %-5level %logger{36}
executing ${sys:cmd} - %msg %n">
            </PatternLayout>
        </Console>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="Console"/>
        </Root>
    </Loggers>
</Configuration>
```

这一段配置了Log4j的行为方式：

- 日志被写入stderr
- ${sys:cmd}会打印到日志里
- %msg是日志消息

对于熟悉log4j的人或者查阅log4j官方文档会发现${java:version}这样的命令用法，因此可以尝试构造`${java:${env:FLAG}}`传入主页的输入框在返回的报错信息第一行得到FLAG。

题解

```
import re
import requests
import sys
import requests
import re,sys

def main(host,port):
    r = requests.post(f"http://{host}:{port}/",data=
{'text':"${java:${env:FLAG}}"})
    print(re.findall("java:(.*?)}",r.text)[0]+"}")


main(sys.argv[1], sys.argv[2])
```

AirticleShare

这道题考察解题者一下技能：

- XSS
- parsley第三方库的利用
- 测信道攻击

尝试题目的功能

题目是个分享文章的论坛，用户在首页提交的文章可以在lookup.php查看，也可以发送给管理员，然后发现管理员就点赞了，不难推断后台是管理员自动化通过访问网页点赞功能出发的，利用Selenium点击`find_element_by_xpath('//input[@id="like"]')`。而flag根据题目题意可能在管理员的某篇文章里。

HTML代码注入

我们做的第一件事是确保我们可以将 HTML 代码注入到文章中。因为题目使用parsley第三方库，验证只发生在客户端，所以一个很自然的想法是，当管理员浏览页面时，我们可以使用 JS 代码或恶意